Networks – Project 1

Implementation of a proxy cache server

# Problem:

As part of the networking course, a group of two people was asked to create a proxy cache server in Java. As a reminder, a proxy server is an intermediate agent that a HTTP client, like a browser, can contact when it needs to request a web site. The caching feature reduces traffic by saving web files for a defined time and returns them to the client as needed. This occurs only if all downloaded data were not modified since the server downloaded them. If not, the server executes a request to the distant web server and saves again files locally.

# Technical constraints:

[REQ\_01]

The proxy cache server shall be started without IDE.

[REQ\_02]

The user shall be able to configure the listening port of the proxy cache server.

[REQ\_03]

The proxy cache server shall support client HTTP versions as 1.0 and 1.1, HTTPS is out of scope.

[REQ\_04]

Only the GET method and static websites shall be tested.

The proxy cache server shall support:

[REQ\_05]

* Get and decode a HTTP request from a client
* [REQ\_06]
* If the requested web page is available locally, the server shall transmit it to the client.

[REQ\_07]

* If the requested web page is not available locally, the server shall forward the request to the web server, get and decode the response, saving files locally and forward the response to the client.

*Note: The solution shall be implemented only with socket handling.*

# Tools and architecture:

The IDE chosen is IntelliJ. IntelliJ IDEA is the most advanced and complete IDE to develop an application in Java. Moreover, the community version is free for a non-commercial use.

The project is hosted on the free and open source distributed version control system GitHub public under <https://github.com/bacalin1982/proxy> .

To implement the solution, the Java SE Development Kit 8 is used.

# Implemented features

## Client-Proxy communication

This feature is at first handled by the Server Class who listen by thread new clients. Then, for each client, in the Client Class, the request is captured by input stream, decoded, logged and transmitted to the cache evaluation. If the cache evaluation is negative or positive, the Client Class is able to give the correct HTTP response to the client.

This communication is adaptive to the client request. Indeed, if the client asks for HTTP 1.0 or 1.1, the server is able to answer.

## Proxy-Server communication

If the cache evaluation is negative, the server is able to forward the client’s request to the distant web server and handle the answer to make new cache files and forward everything to the client.

Competing connections

This feature is implemented by accepting clients with thread method in Server Class. The server is able to handle multiple connections simultaneously.

## Pipelining

Not implemented yet.

## Persistence

## Caching

## Saving cache

# User guide

The server is runnable without IDE. At least one argument is mandatory to start the application. Use: Proxy <server\_port>. If the first argument is not a number, a default 8081 port will be set instead.

The application was tested with following web sites:

* [www.perdu.com](http://www.perdu.com)
* [www.meme.com](http://www.meme.com)
* <http://httpwg.org/specs/>
* <http://httpd.apache.org/docs/2.2/fr/>